University of Applied Sciences Bremen

Bachelor Thesis

Efficient Coding of Information in Neurobiologically Motivated Deep Network Architectures

Modelling Peripheral Vision with Convolutional Neural Networks

Noshaba Cheema

Supervised By

Prof. Dr. Martin Hering-Bertram
Dr. Christoph Zetzsche

14^{th} May, 2016

Preface

Abstract

Recent studies have shown that the loss of spatial acuity and permutations (so called "crowding-effect") in the visual periphery can be modelled using local summary-statistics. These statistical measures are based on auto- and cross-correlations, which operate on wavelet-like filter outputs. However, the neurobiological plausibility of the multiplications which are required for these correlations are debated. We investigate, whether a more biologically model can be developed by using the filter outputs of a deep neural network and neurobiologically realistic AND-like operations instead of a multiplication operation. We show that our model yields a similar amount of information about the image as the formal approach. Furthermore, we investigated which level of of the visual system might provide the best statistical representations. Common assumption about this suggests that the features to be statistically pooled should be relatively low level and simple. However, it is unclear why such low-level features should be best suited for such a representation. In order to investigate which level of the visual system might be best suited, we used a high-performing convolutional neural network to model the latter and reconstructed statistical representations from solely one layer and found an optimum in the intermediate layers.

Acknowledgements

This thesis was done with the Cognitive Neuroinformatics group at the University of Bremen under the supervision of Christoph Zetzsche. Whom I would like to thank the most for giving me such an opportunity to work on such a current research topic. Furthermore, I would like to thank Kerstin Schill for believing in me, Konrad for enduring my problems with the computer and Simon Bjerring for all the cat pictures.
For creative input I would like to thank Udo Frese, Ruth Rosenholtz, Lex Fridman, Leon Gatys, Konrad Gadzicki, Tobias Kluth, Martin-Hering Bertram, Thomas Jahn, Soumen Ganguly, Mostafa Ahmed Abdelfattah and Tobias Schemken.
Finally, a big thank you to Tobi S. for staying by my side.

Contents

Chapter 1

Introduction

Our senses are confronted with more information than we can process. This becomes apparent in the human peripheral field of view. Peripheral vision gives us a worse representation of the visual input than the foveal vision [1, 32], which is located in the center of the retina. Information seems to appear coarser in the periphery and local permutation effects - the so-called "crowding effect" - happen. Crowding is called the phenomena in which object recognition is affected in the visual periphery, when neighbouring objects (so-called "flankers") coexist. It is hypothesized that the crowding effect is a result of dealing with a bottleneck of visual sensory information.
There have been models introduced by Rosenholtz [32] and Freeman [9] which simulate these effects using local summary statistics based on V1-like filter outputs. Our model introduces a new variant of such models by using feature maps generated by a high-performing convolutional neural network and by introducing neurobiologically motived statistics, which make no use of specific multiplication operations. Furthermore, we investigate which layer of a convolutional neural network provides the best statistical representation.
Convolutional neural networks have challenged the human visual system

when it comes to object classification and object recognition tasks [36]. We thus examined whether the efficient coding of features in such hierarchical networks can be used in order to model peripheral crowding.

This work is divided into two main parts. Chapter 2 to 4 deal with theoretical background of machine learning and peripheral vision and chapter 5 introduces our approach to this problem.

Chapter 2

Machine Learning

The following chapter explains the basics of Machine learning and the various types of classification models associated with it. It can be defined as the ability of a program to learn from experience − that is, to modify its execution on the basis of newly acquired information. This is often a very attractive alternative to constructing them, and in the last decade the use of machine learning has spread throughout computer science and beyond. Machine learning is used in Web search, spam filters, recommender systems, ad placement, credit storing, fraud detection, and many other applications. There are several fine textbooks available to gain the insights in machine learning (e.g [24]).

2.1 Unsupervised Learning

In Unsupervised learning, the goal is to have the computer learn how to do something without explicitly telling it how to do it. There are two approaches to unsupervised learning. First, is a form of reinforcement learning, where the agent bases its actions on the previous rewards and punishments without

necessarily even learning any information about the exact ways that its actions affect the world. In a way, all of this information is unnecessary because by learning a reward function, the agent simply knows what to do without any processing because it knows the exact reward it expects to achieve for each action it could take. This can be extremely beneficial in cases where calculating every possibility is very time consuming (even if all of the transition probabilities between world states were known). On the other hand, it can be very time consuming to learn by, essentially, trial and error [38]. And, a second type of unsupervised learning is called clustering. In this type of learning, we assign a set of subsets(called clusters) so that observations in the same cluster are similar in some sense.

Unsupervised learning differs from supervised learning in that the learner is given only unlabeled samples.

2.2 Supervised Learning

Every instance in any dataset used by machine learning algorithms is represented using the same set of features. The features may be continuous, categorical or binary. If instances are given with known labels(the corresponding correct output) then the learning is called supervised learning. Supervised learning along with other classification techniques is well illustrated by Kotsiantis.

A supervised learning algorithm analyses the training data and produces an inferred function, which is called a classifier(if the output is discrete) or a regression function(if the output is continuous). A classifier implements classification, which is an example of a general problem of pattern recognition

that assigns some sort of output value to a given input value. A regression function on the other hand implements regression, which assigns a real-valued output to each input value. Examples of regression include part of speech tagging, parsing etc.

There exists a large number of algorithms for classification which have a linear function that assigns a score to each possible category of input. One such algorithm is a Support Vector Machine(SVM), which given a set of training examples, each marked for belonging to one of two categories(Classification and Regression), builds a model that assigns new examples into one category or the other. Another common model are neural networks, which will be described in the following chapter in more detail.

Chapter 3

Neural Networks

Artificial Neuronal Networks (ANNs) are subject of the field of Computational Neuroscience. Inspired by biological neuronal systems, primarily the human brain, they are networks of multiple complex layers of non-linear transformations to process information or for modelling [31].
In the recent years there has been a boom using ANNs for classification tasks. This is due to the human-challenging performance in classification and recognition tasks [36], which has been made possible due to large sets of labelled data being publicly available and due to the recent development in GPU-acceleration [34].
ANNs are not only used for classification purposes however. There have been applications from making them "dream" [25], to generating image descriptions [18], texts [19], music [16] or art [11]. Due to their high performing qualities and wide range of possibilities to use them, it is no surprise why they have become so popular.

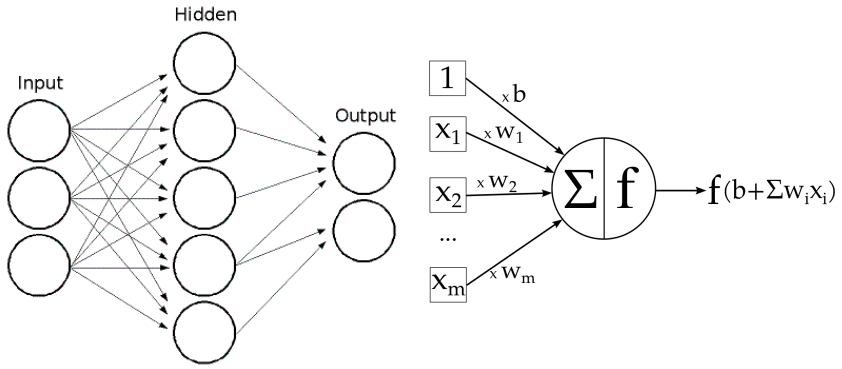


Figure 3.1: *Left*: Basic Artificial Neuronal Network with one hidden layer. *Right*: Artificial Neuron (x_i: Input, b: bias, w_i: weights, f: activation function).

3.1 Basic Architecture

An ANN consists of multiple connected neuron layers. The layers between the input neuron layer and the output layer are called hidden layers (Fig. 3.1 *left*). These neurons have an activation function, which receives a weighted sum as input (Fig. 3.1 *right*) and forwards the "activated output" to the next layer. This "activated output" could for example mean something like "Thing X might be to 10% a fork and to 90% something else". The process of forwarding this output to the next layer of neurons is called *Forward-Propagation*. Usually, at the end of the Forward-Propagation a classification result is obtained. To keep the error in the classification result as low as possible, the network must be trained with labelled data. In this process, the data first gets forward-propagated. Afterwards, utilizing a loss function, the difference between the classification result and the data's original label is examined. This loss function could be the mean squared error, for example. Then, using *Back-Propagation*, the weights in the network get augmented in such way that the loss function is minimized. This is usually done by using gradient decent. Furthermore, the output of each layer is called a *Featur-*

Map. [31]

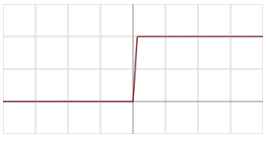
3.2 Activation Functions

In neural networks the activation function is an abstract representation of the rate of action potential firing in a neuron. In the simplest case this can be a binary ON/OFF-function. Due to its non differentiable properties, the most common activation function used is one of sigmoidal shape. A function of sigmoidal shape is also biologically the most plausible one, since it has some properties of biological neurons. The neuron's firing frequency stays zero until input current is received, then quickly increases at first, until it gradually approaches an asymptote at maximum firing rate.
Since neural networks should be able to handle complex problems, the activation function should be a non-linear function. [31]

This is a collection of commonly used activation functions:

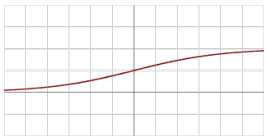
Binary Step

$$f(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$$



Sigmoid

$$f(x) = \frac{1}{1 + e^{-x}}$$



TanH

$$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$$

Rectification Linear Unit (ReLU)

$$f(x) = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$$

3.3 Forward-Pass

The forward-propagation or *Forward-Pass* is illustrated in figure 3.2. The input x is visualized as the first activation $a^(1)$ in the figure. It then gets weight with the weight matrix $\Theta^(i)$ in layer i. Since every neuron has a connection to every neuron in the next layer (except for the bias neuron (+1)), one can write the weights as a matrix and make use of accelerated matrix multiplications. The weight activation $z^{(i)}$ is then used as an input for the activation function g in the next layer, which then produces another activation $a^{(i+1)}$. This gets then forward propagated to the next layer and so on, until you get to the classification result $h_\theta(x)$. Forward-propagation can be referred to as "testing" the network. [31]

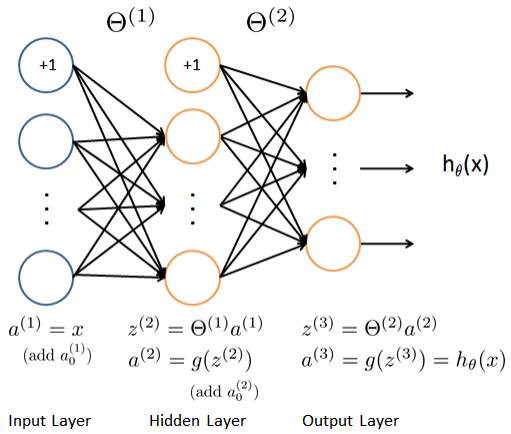


Figure 3.2: Forward-Pass in a Neural Network (x = input, $a_j^{(i)}$ = activation, $\Theta^{(i)}$ = weight matrix), $g(z)$ = activation function, $h_\theta(x)$ = classified output)

3.4 Backward-Pass

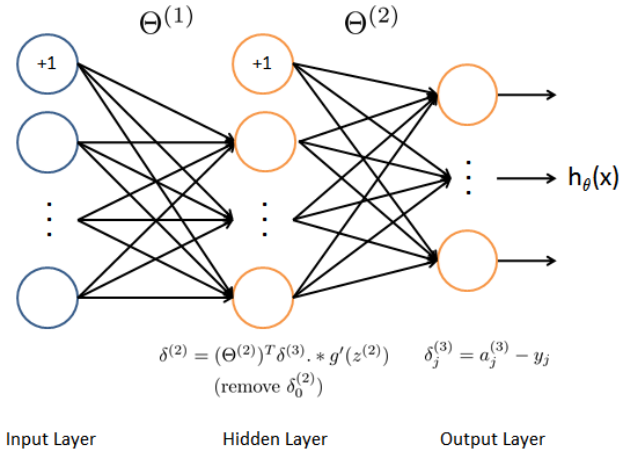


Figure 3.3: Backward-Pass in a Neural Network (y_j = original label, $a_j^{(i)}$ = activation, $\Theta^{(j)}$ = weight matrix), $g'(z)$ = derivative of activation function, $\delta_j^{(i)}$ = error)

Fig. 3.3 illustrates the backward-propagation or *Backward-Pass*. In the backward-pass the weights get adjusted, so that the difference between the original label y and the computed label from the network $h_\theta(x)$ is very small or zero. In Fig. 3.3 the computed label is referred to as $a_j^{(3)}$, which just

means $h_\theta(x)$ is the resulting activation from the last layer. The error $\delta^{(i)}$ that is computed from the difference of y and $a_j^{(3)}$ is then back-propagated to the previous layer and is weight with the transpose of the weight matrix $\Theta^{(i-1)}$ and element-wise multiplied with the derivative of the activation function g. You do this until you reach the first layer. In order to compute the new weights, one can use gradient descent to find new parameters $\Theta^{(i)}$, which satisfy that the error δ is gradually shifted towards 0 or at least a local minimum. Backward-propagation can be referred to as "training" the network. [31]

3.5 Types of Neural Networks

This section describes briefly selected, commonly used neural networks.

3.5.1 Feed-forward Networks

Feed-forward networks are neural nets that only have connections to the next layer and no connections to the previous layer or connections within the layer itself. They are the simplest and most commonly used neural networks, since they are easily trainable. Fig. 3.1 illustrates a feed-forward network.

3.5.2 Recurrent Neural Networks

Recurrent Neural Networks (RNN) are networks that allow loops and connections towards prior layer or connection within the current layer in addition to the connections to the next layers. They are commonly used in speech recognition [14] but are significantly harder to train.

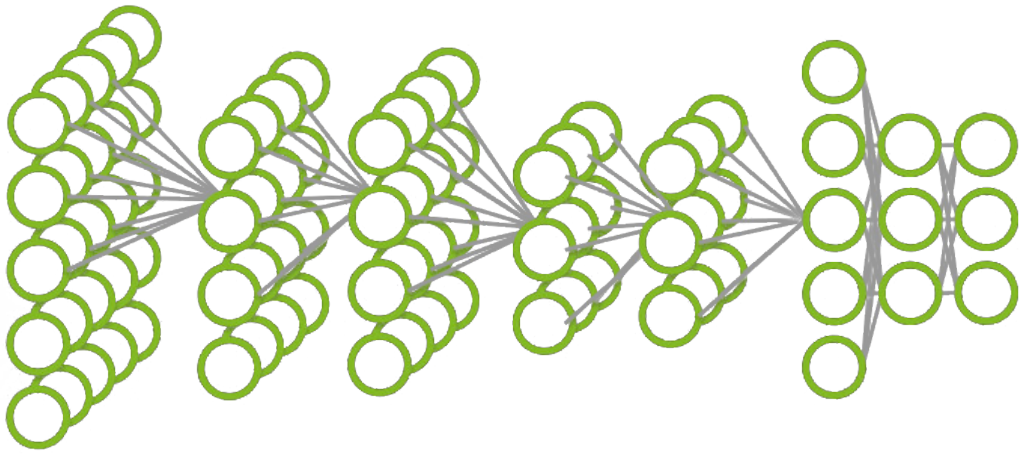


Figure 3.4: Convolutional Neural Network

3.5.3 Convolutional Neural Networks

A classical ANN consists of *fully-connected* layers, where every single output from the previous layer gets forwarded to every single neuron in the next layer as input. The general definition of a Convolutional Neural Network (CNN) on the other hand does not state that every single neuron needs to forward its output to every single neuron in the next layer (Fig. 3.4). Usually a discrete *convolution*, which gives the network model its name, is implemented, however. A convolution calculates a weighted sum over a specific window. The size, or the elements, of the window is defined by the convolution kernel. Then the window is moved on by a specific distance (*Stride*) until the whole input of the layer got covered. This type of neural network is commonly used for image classification.

Layer Catalogue

The layer catalogue explains briefly the most important layers in a CNN.

Convolutional Layer

The convolutional layer in a CNN *convolves* the activations of the previous layer before the activation function is applied on them in the current layer. *Convolving* means that a weighted sum over a specific window is calculated. The weights are determined by the convolution kernel. In a convolutional neural network this kernel is mostly a matrix of size $N \times N$. The convolution kernel then moves from pixel to pixel, with the distance given by the stride, and weights every underlying pixels with its coefficients and sums over those. This is continued until the whole image is convolved. These convolutions calculated specific features such as edges or orientations.

Eq. 3.1 shows the formal definition of the discrete convolution. f is here the feature map from the previous layer and g the convolution kernel.

$$(f * g) = \sum_{m=-\infty}^{\infty} f[m] - g[n - m] \tag{3.1}$$

The biological idea behind convolving the input from the prior layer is that the convolution kernel can be seen as a receptive field of a neuron in the brain, which perform basically the same actions. Interestingly, when training a CNN with an image data set, which is general enough, Gabor-filters start to develop in the first layers. These Gabor-filters have also been found in the primary visual cortex of the human brain. A CNN can thus be seen as a simulation of the human visual system.

ReLU Layer

A ReLU (Rectified Linear Unit) layer performs an element-wise activation function to its input, such that $max(0, x)$. With this the problem of the

vanishing gradient is counteracted. Since most networks use a sigmoidal activation function which has the range $[0, 1]$, the gradients tends to go towards 0 and less changes to the weights can be done, especially for deep neural networks. The ReLU layer allows activations with a range greater than 1.

Pooling Layer

The pooling layer subsamples the activations to its most significant features by averaging or taking the maximum over a given windows size. This window then moves with the distance given by its stride, similar to the convolution kernel, to the next pixel until the whole picture has been pooled.

Convolution and pooling are usually applied in an alternating manner.

Fully-Connected Layer

A fully-connected layer is a layer where every neuron from the current layer is connected to every neuron in the next layer, as seen in regular neural networks. In convolutional neural networks this done at the end after all convolution and pooling layers. It computes the class scores in the end.

Chapter 4

The Visual System

4.1 The Visual Cortex

The visual cortex, located at the back of the skull on both hemispheres, is the region of the brain associated with analysing, interpreting, and memorizing the information perceived by the neurons in the Retina. It can be divided into six different Visual areas, V1 to V6, each area responsible for different tasks in the processing and amplifying of the visual information perceived. The model for the process of analysing and recognising visual input further used here is the dorsal/ventral model, which describes the processing of visual input in the brain as a two pathway system. The information reaches the V1 area unfiltered and is than split into two streams, the dorsal and the ventral stream. In the early 1980's, L.G. Ungerleider and M. Mishkin were the first to describe the what vs. where account of those two pathways. That is

1. The *dorsal stream* starts from the V1 area, through the V2 area and the V6 and V5 area and ends up in the posterior parietal cortex which is responsible, among other tasks, for spatial reasoning and planned movements. This path is referred to as the *Where Pathway* or the *How*

Pathway.

2. The *ventral stream* starts from the V1 area, through the V2 and V4 area to the inferior temporal cortex. This path is referred to as the *What Pathway* and associated with recognition of form and colour, object representation, and the forming of long-term memory. [39]

M.A. Goodale and A.D. Milner, in the early 1990's, again pointed out that neuropsychological, electrophysiological and behavioural evidence suggests that the neural substrates of visual perception may be quite distinct from those underlying the visual control of actions, and proposed that the ventral stream of projections from the striate cortex to the inferotemporal cortex plays the major role in the perceptual identification of objects, while the dorsal stream projecting from the striate cortex to the posterior parietal region mediates the required sensorimotor transformations for visually guided actions directed at such objects. [13]

4.1.1 The visual areas

Now we add some more information on the visual areas most relevant to this thesis:

V1

The V1 area is the first, most primitive, and most studied area of the visual cortex. It can be found in all mammals. All visual information first gets progressed in the V1 area, mostly responsible for pattern recognition. All visual information reaches the V1 area completely unfiltered, even including the blind spots in the retinas. Besides processing visual information from the retinas, the V1 area also processes feedback from the higher level visual

areas. Especially the pattern recognition properties of the V1 area give it a central roll in the computational work described further on. [9]

V2

Cells in V2 are tuned to simple operations as in V1, but also to more complex properties as illusory contours, binocular disparity, and whether the stimulus is part of the figure or the ground. The V2 area is strongly connected to the higher level areas (V3 to V5) and sends strong feedback to V1 as well.

V3

This section is mainly part of the dorsal stream but also has some weak connections to the other areas of the ventral stream.

V4

This area my be further seperated into four sub-areas. In general V4 neurons are receptive to a number of properties, such as color, brightness, and texture. It is also involved in processing shape, orientation, curvature, motion, and depth. The V4 area was first found in macaque monkeys and later in humans as well. It is still of interest in research. [30, 2]

V5

The V5 area is mainly concerned with a moving stimulis speed and its direction of movement. Since this is also carried out by the V1 area, there is still some controversy about what influence the computations done in V5 have. It is proposed that V5 neurons are tuned for more complex stimuli than their V1 counterparts. [26]

4.2 Peripheral Vision

Peripheral vision is non-central vision. It is achieved by light rays falling on the retina outside the macula, which is situated in the center of the retina. Peripheral vision is the result of these light rays stimulating the rods. Therefore, as in night vision, the sharpness of peripheral vision is weak and color perception is not strong. Weak sharpness can also be attributed to the fact that most of the light sensors lie in the middle of the retina and they decrease far from the center. [37]

4.2.1 Visual Crowding

Visual crowding is a property of peripheral vision. It means that the shape of an object in the peripheral vision cannot be recognized as well when it is flanked by other objects. The crowding effect depends on the distance between the object and other nearby objects (flankers) and the arrangement of those flankers with respect to the object. One theory to explain crowding is that it is a form of averaging between of the object and the flankers, which can be understood as a form of blurring. Moreover, crowding can be linked to texture perception. Since crowding depends greatly on the similarity between the object and the flankers, making it harder to distinguish the object when the flankers are similar to it, we can deduce that peripheral vision employs some kind of texture recognition technique. This texture recognition technique results in the inability to distinguish similar objects when they are next to one another, lumping them together in one texture. The crowding-effect can be seen as a strategy of the visual system to deal the information bottleneck in sensory processing. As such, it may also be of interest for image compression, feature extraction and computer vision. [37]

Chapter 5

Modelling Peripheral Vision with CNNs

The following chapter deals with implementing a model for simulating the crowding effect in the visual periphery using convolutional neural networks. The crowding effect shows that the reduction of information in the periphery is not just a reduction of the resolution. One method of modelling such representation was explained by Rosenholtz [32] by using local summary statistics in order to simulate the local texture-like permutations to create images that show the crowding effect on the sides of the image and a sharp central foveal region in the center of fixation. The images were named *Mongrels* by her. Freeman and Simoncelli used a variant of this in order to create similar images, which they called *Metamers* [9].

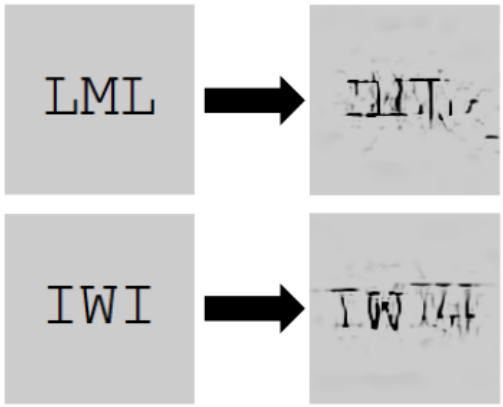


Figure 5.1: *Mongrels* reconstructed from a model of Rosenholtz [32]

That statistical covariances are well suited for generating textures has been already shown by Portilla and Simoncelli [29], which has been described as one of the best performing parametric models for texture-synthesis so far [12]. It is based on a set of carefully hand crafted summary statistics computed on the responses of their linear filter bank called *Steerable Pyramid* [35], which resembles the filters in the early visual system of primates.
In order to restrict the summary statistics to local regions and divide an image in an foveal and a peripheral region, Rosenholtz used a Texture-Tiling Model (TTM) [32]. It subdivides the image in local, overlapping pooling regions. These pooling regions grow with eccentricity, the further they are away from the center of fixation (Fovea) [32]. The summary statistics of Simoncelli and Portilla are then computed within these pooling regions. The TTM resembles, in a way, the receptive fields of the visual system [9].

Instead of using the small feature space of Portilla and Simoncelli, which is based on the early visual system, this work uses the large feature space provided by a high-performing convolutional neural network, which is a functional model for the entire ventral stream, as non-linear filter banks to com-

pute such summary statistics. A similar approach has been done by Gatys et al. for generating textures [12] and transferring the style of one image to another [11]. However, instead of using the feature space of every level up to a specific one, we test which layer is best suited for such a statistical representation and only model peripheral vision from this layer. Furthermore, our model makes use AND-like non-linearities, instead of multiplication operations, due to the debate within researchers, whether such operations are neurobiologically plausible [20]. We show that these AND-like operations deliver similar results to a formal multiplication [4].

5.1 Caffe

BLVC Caffe [15] is a C++ framework by the *Berkeley Vision and Learning Center* to design, implement, train and test neural networks. It is widely used within professional research and applications [11, 12, 25, 36], since it is open-source, uses GPU-accelerated code for its default layers and has interfaces for Python and Matlab for high-level evaluations. Many trained neural networks have been made publicly available as *caffemodels* which is another great advantage for using this framework.
A trained caffemodel of the VGG-19 net [36] by Simonyan and Zisserman has been used for this work.

5.2 VGG-19 Layer Net

The VGG-19 network was developed by Simonyan and Zisserman from the Visual Geometry Group at the University of Oxford [36]. It is a on object recognition trained convolutional neural network with 19 weight layers. The

architecture of this network was inspired by Krizhevsky et al. [21] and Ciresan et al. [6]. With their configuration they scored ranks 1 and 2 at the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) 2014 in the categories *Localisation* and *Classification*.

5.2.1 Architecture

The net contains 16 convolutional, 3 fully-connected and 5 pooling layers. The convolution kernels just had a size of $3 \times 3 \times k$ each, where k is the number of input feature maps. Stride and padding were chosen to be 1, so the output feature maps have the same spatial dimensions as the input feature maps. For pooling the net uses max-pooling with non-overlapping pooling regions of size 2×2. Hence, the feature maps get down-sampled by a factor of 4 to the most dominant features. Similar to AlexNet [21], the VGG-19 net uses linear rectification units, instead of a sigmoid function, for the activation function, in order to get a faster convergence time. [36]

Layer	Conv/Pool-Info	Feature Maps	Output Size	Spatial Resolution
conv1_1	K-size: 3, Pad: 1	64	$64 \cdot N \cdot M$	$N \cdot M$
conv1_2	K-size: 3, Pad: 1	64	$64 \cdot N \cdot M$	$N \cdot M$
pool1	K-size: 2, Stride: 2	64	$16 \cdot N \cdot M$	$\frac{1}{4} \cdot N \cdot M$
conv2_1	K-size: 3, Pad: 1	128	$32 \cdot N \cdot M$	$\frac{1}{4} \cdot N \cdot M$
conv2_2	K-size: 3, Pad: 1	128	$32 \cdot N \cdot M$	$\frac{1}{4} \cdot N \cdot M$
pool2	K-size: 2, Stride: 2	128	$8 \cdot N \cdot M$	$\frac{1}{16} \cdot N \cdot M$
conv3_1	K-size: 3, Pad: 1	256	$16 \cdot N \cdot M$	$\frac{1}{16} \cdot N \cdot M$
conv3_2	K-size: 3, Pad: 1	256	$16 \cdot N \cdot M$	$\frac{1}{16} \cdot N \cdot M$
conv3_3	K-size: 3, Pad: 1	256	$16 \cdot N \cdot M$	$\frac{1}{16} \cdot N \cdot M$
conv3_4	K-size: 3, Pad: 1	256	$16 \cdot N \cdot M$	$\frac{1}{16} \cdot N \cdot M$
pool3	K-size: 2, Stride: 2	256	$4 \cdot N \cdot M$	$\frac{1}{64} \cdot N \cdot M$
conv4_1	K-size: 3, Pad: 1	512	$8 \cdot N \cdot M$	$\frac{1}{64} \cdot N \cdot M$
conv4_2	K-size: 3, Pad: 1	512	$8 \cdot N \cdot M$	$\frac{1}{64} \cdot N \cdot M$
conv4_3	K-size: 3, Pad: 1	512	$8 \cdot N \cdot M$	$\frac{1}{64} \cdot N \cdot M$
conv4_4	K-size: 3, Pad: 1	512	$8 \cdot N \cdot M$	$\frac{1}{64} \cdot N \cdot M$
pool4	K-size: 2, Stride: 2	512	$2 \cdot N \cdot M$	$\frac{1}{256} \cdot N \cdot M$
conv5_1	K-size: 3, Pad: 1	512	$N \cdot M$	$\frac{1}{512} \cdot N \cdot M$
conv5_2	K-size: 3, Pad: 1	512	$N \cdot M$	$\frac{1}{512} \cdot N \cdot M$
conv5_3	K-size: 3, Pad: 1	512	$N \cdot M$	$\frac{1}{512} \cdot N \cdot M$
conv5_4	K-size: 3, Pad: 1	512	$N \cdot M$	$\frac{1}{512} \cdot N \cdot M$
pool5	K-size: 2, Stride: 2	512	$\frac{1}{2} \cdot N \cdot M$	$\frac{1}{1024} \cdot N \cdot M$
3× Fully-connected layer				
Softmax				

Table 5.1: VGG-19 Network Architecture

The linearly rectified convolutions and the max-pooling are applied in an altering manner as seen in Table 5.1. Until *pool*4 the feature maps are doubled while the spatial resolution quarters (Table 5.1). N and M are the input dimensions.

5.2.2 Normalization and Average Pooling

This implementation uses a different version of the VGG-19 net from Gatys et al. [11, 12]. In contrast to the original VGG-19 net which uses max-pooling, the pooling type here was changed to average-pooling, which improves the gradient flow and one obtains slightly cleaner results. For practical reasons the Gatys et al. version of the VGG-19 layer net has been normalized by scaling the parameters such that the average activation of each filter is equal to 1 [12]. Furthermore, the fully-connected and the softmax layers have been removed.
The normalized version of the network can be downloaded from
`bethgelab.org/media/uploads/deeptextures/vgg_normalised.caffemodel`.

5.3 Summary Statistics Model

To model the texture-like permutations for the crowding-effect, this work makes use of a summary statistics model, similar to Rosenholtz [32, 33] and Freeman and Simoncelli [9]. These models make use of the formal definition of statistics in mathematics. Thus, these models are called *Mathematical Models* (in spite of its other neurobiology-related components) in the following pages. Since such multiplication operations have been debated [20], this work makes use of operations, which are more biologically plausible. Thus, this model is

being referred to as the *Neurobiological Model* in the following.
The comparison between these two models is done in the section "Modelling Peripheral Vision", however.

5.3.1 Mathematical Models

Rosenholtz suggested that crowding can be modeled as statistical spatial pooling of visual features computed in V1 [1, 33] and developed an appropriate model [1] by making use of an algorithm for texture synthesis [29]. Freeman and Simoncelli used a variant of this model to describe the computations in the ventral stream and the resulting perceptual *Metamers* [9].
These statistical representations are computed by directly replicating the formal statistical definitions. For example, by making use of averages or correlations of the feature maps.
An empirical cross-correlation of the feature maps A and B is computed as:

$$C[A(\Delta s), B(\Delta t)] = \sum_{s,t} A(s + \Delta s)B(t + \Delta t) \tag{5.1}$$

This makes explicit use of the *multiplication* operation.
However, researchers have long debated whether such operations are neurobiologically possible [20]. The formal statistical correlations might not even be necessary, since much of the functionality may be preserved if one replaced the multiplication operation with one, which is more biologically plausible.

5.3.2 Neurobiolocal Model

A function that exhibits similar characteristics to the multiplication operation is an AND-like computation, as suggested by Zetzsche and Barth [40]. Since biological hardware can easily realize an ON/OFF rectification and

non-linear transducer functions with sigmoid shape, one can make use of an old Babylonian trick to derive a suitable AND-like computation:

$$AND(a, b) = N[a + b] - N[a - b] \tag{5.2}$$

where N is a suitable non-linear transducer function and a and b feature maps.

These AND operations are characterized by the property that they attain a kind of local "maximum" (for the sum $a + b$ constrained) if a and b have the same size. If a or b is decreased, the response is systematically reduced, until it vanishes if either a or b equals zero. If N is a sigmoid non-linearity the resulting AND will have a threshold-like behavior for small input values and will go into saturation for large input values. Hence, a multiplication can be considered as a special case of this generalized AND-function.

This type of AND-operation may be, neurobiologically, the most plausible one.

5.4 Texture Tiling Model

Similar to prior models of generating images that are peripherally distorted in a texture-like fashion [32, 33, 9], this work makes use of a Texture-Tiling Model in order to model receptive fields.

This TTM consists of local regions (called *pooling regions*) in which the required statistical representations are computed. These pooling regions grow from the center of fixation of the field of view, hence there are more permutations in the peripheral field of view than in the center. For simplicity, the pooling regions for the TTM were implemented as rectangular regions, instead of elliptical or round ones. Similarly to prior methods [32, 9], these

pooling regions, overlap each other, though.

5.4.1 Methods

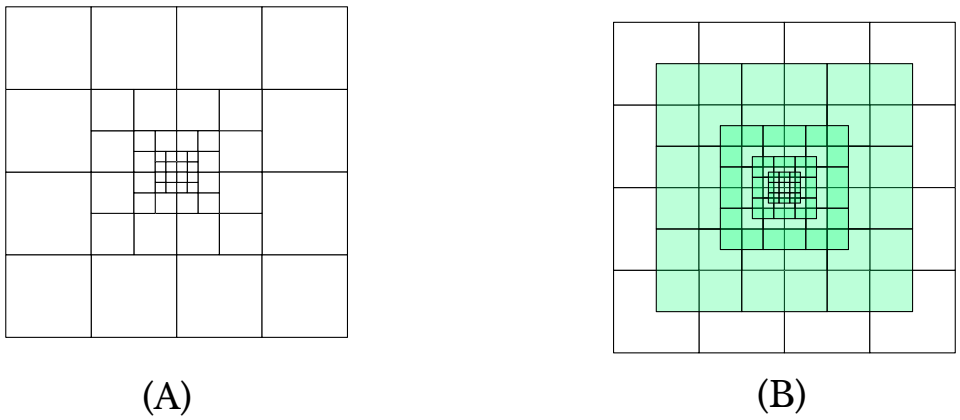


Figure 5.2: Texture-Tiling Model. *Left*: Non-overlapping pooling regions *Right*: Overlapping pooling regions

First, non-overlapping pooling regions are computed over the image, which become smaller from the edge to the center of the image by a factor of 0.5 in each direction (Fig. 5.2 (A)). Research in visual crowding suggests that these pooling regions grow linearly with eccentricity from the center of fixation of the sight, with a radius of a radius of approximately 0.4 to 0.5 the eccentricity. This has been dubbed as "Bouma's Law" [28, 37].
After that, for each size of a pooling region, an overlapping layer of pooling regions is added to the Texture-Tiling Model (Fig. 5.2 (B)). This way we allow a better flow between neighbouring pooling regions.
The neurobiological statistics are then computed within these pooling regions.

5.5 Optimum Layer

Another hyper-parameter that is worth considering, while making use of the feature maps of a CNN, is from which layer(s) these feature maps should be obtained in order to model a statistical representation.

The first layers of a convolutional neural network resemble the primary visual cortex with low-level features. Filter kernels that are computed within these layers correspond to Gabor filters, which have also been found the in the V1-area of the visual cortex [3].

Research on human vision has recognized such a statistical pooling to be a crucial factor for the representation of information in the peripheral field of view, which accounts about 98% of the field of view [1, 9].

The common assumption is that low-level features are best suited for such a statistical pooling [29, 32]. Models that make use of statistical representations of features, such as *Histogram of Oriented Gradients* (HOG) [7], *Scale-Invariant Feature Transform* (SIFT) [23], *Textons* [17] or models that represent crowding [1, 9, 32], make use of the feature space provided by V1-like filters.

As convincing as these applications may appear, it is not clear why such representation needs to be extracted from the early level of local oriented features or − in neurobiological terms − why such pooling should act directly after the primary visual cortex [5].

Since the visual system itself consists of a multi-level hierarchy and visual models that are based on hierarchical architectures, such as CNNs, have proven to be human-challenging when it comes to classification and recognition tasks [36], it makes sense to assume that such hierarchical architectures are crucial for obtaining an efficient representation. However, it is far from obvious which level of such a hierarchical architecture is best suited for this

task [5].

This chapter investigates this by comparing statistical representations reconstructed from different levels of a hierarchical architecture which represents the visual system, such as CNNs, and observe which level is best suited for a texture-like or statistical representation. For the CNN we make use of the VGG-19 net [36].

5.5.1 Methods

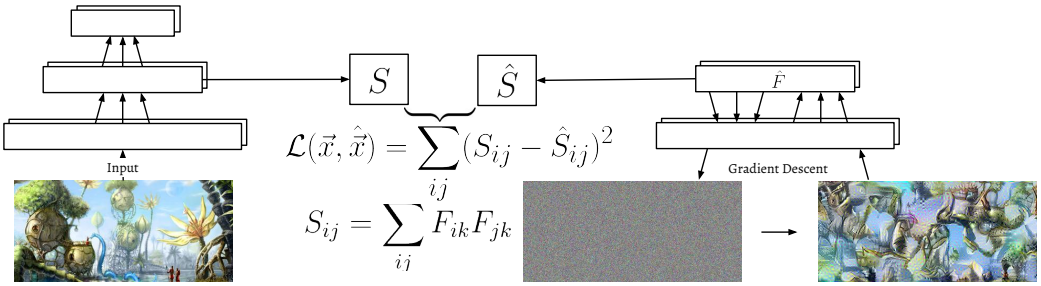


Figure 5.3: Schematic architecture of the deep network and reconstruction algorithm. (F_{ik}: feature i at position k; S_{ij} : covariance of features F_i, F_j) [5]

The method that is used here for a statistical representation is much of the spirit of Gatys et al. They used their statistical representation of the feature space of a neural network for generating textures [12]. With their model they outperformed the texture-generation model by Portilla and Simoncelli [29], which had been described as one of the best parametric models [12] for 15 years.

To generate a texture from a given source image $\vec{x}$, different features a extracted from this image by using the forward pass of a CNN. To obtain a stationary description of the image, summary statistics are computed on the feature responses of the image. Next a new image with the same stationary description, and thus with the same spatial summary statistics, is computed

by performing gradient descent on random white-noise image (Fig. 5.3). [12, 11]

However, instead of using all the layers up to a specific layer to compute the feature responses like Gatys et al., this work makes use of the feature space of just one layer.

Synthesis

At first, one needs to compute the feature responses of an image using a CNN by forwarding it to the network. The filter kernels of the trained neural network are then applied to the image and one obtains the feature responses (so called *feature maps*) [31].

A layer with N_l kernels has N_l feature maps of the size M_l, if vectorized. One can save these feature maps in a matrix $F^l \in \mathbb{R}^{N_l \times M_l}$, where F^l_{jk} is the activation of the filter j at position k in layer l. To generate a texture from this feature map matrix, the spatial information of the original image has to be discarded. A statistic that has such a property, is given by the correlation of two feature maps of a layer. [12]

These correlations are given by the Gramian matrix $S^l \in \mathbb{R}^{N_l \times N_l}$, where S^l_{ij} is the inner product between feature maps i and j in layer l [12]:

$$S^l_{ij} = \sum_k F^l_{ik} F_{jk} \tag{5.3}$$

Note that this statistical representation also makes specific use of multiplication operations. However, since we just want to determine which layer is best suited for any statistical representation, we can exploit the fast computation time of an inner product, in order to generate quick results for comparison. Furthermore, since we just want to use one layer for texture synthesis, l can

be disregarded.

To generate a new texture on the basis of the original image, one can use gradient descent on a white noise image to find a corresponding image with the same (or similar) Gramian matrix − and thus the same statistical representation − as the original image's Gramian matrix. For the loss function this work uses the mean squared error between the statistical representation S of the original and of the statistical representation $\hat{S}$ of the generated image:

$$\mathcal{L}(\vec{x}, \hat{\vec{x}}) = \frac{1}{4N_l^2 M_l^2} \sum_{ij} (S_{ij}^l - \hat{S}_{ij}^l) \tag{5.4}$$

$\vec{x}$ is the vectorized original image and $\hat{\vec{x}}$ the vectorized generated image.

The gradient of the function with respect to the activations $\hat{F}^l$ can then be computed analytically:

$$\frac{\partial \mathcal{L}}{\partial \hat{F}_{ij}^l} = \begin{cases} \frac{1}{N_l^2 M_l^2} \left((\hat{F}^l)^{\mathrm{T}} (S - \hat{S}) \right)_{ji} & \text{if } \hat{F}_{ij}^l > 0 \\ 0 & \text{if } \hat{F}_{ij}^l < 0 \end{cases} \tag{5.5}$$

The gradient $\frac{\partial \mathcal{L}}{\partial \hat{\vec{x}}}$ with respect to the image $\hat{\vec{x}}$ can then be computed via the standard back propagation of the neural network [22], which can then be used as an input for a numerical optimization strategy. This work uses the L-BFGS-B solver [42], which seems to be a reasonable choice for such a high optimization problem.

Since the whole procedure relies on the standard forward-backward pass of the Caffe framework [15], whereby the advantages of the GPU acceleration can be utilised, these operations could be computed in a reasonable time.

For comparison, this reconstruction procedure was also applied to the "raw"

feature maps F_{ik} of the corresponding layer, where we directly reconstruct from the coefficients in a single layer of the hierarchy. Whereas, in the case of the statistical representation, the new image is reconstructed from the Gramian matrix of those feature maps.
The loss function for the raw feature reconstruction is as follows:

$$\mathcal{L}(\vec{x}, \hat{\vec{x}}) = \frac{1}{2}\left(F_{ij}^{l} - \hat{F}_{ij}^{l}\right)^{2} \tag{5.6}$$

The derivative of this loss with respect to the activations in layer l equals:

$$\frac{\partial \mathcal{L}}{\partial F_{ij}} = \begin{cases} \left(F_{ij}^{l} - \hat{F}_{ij}^{l}\right) & \text{if } F_{ij}^{l} > 0 \\ 0 & \text{if } F_{ij}^{l} < 0 \end{cases} \tag{5.7}$$

5.5.2 Results

The results in Fig. 5.5 show that the first layers reconstruct the best results for the "raw" features, which is not that surprising, due to the systematically decreased amount of information by the increasing abstraction towards higher stages of the architecture (Fig. 5.4 upper and middle rows). The reconstructions become more and more distorted towards higher levels. [5]
Interestingly, the same effect is not observed for the reconstruction from the statistical representation (Fig. 5.4 bottom row). Rather, the reconstruction quality is low at the initial stage, then gradually increases up to the intermediate layers, and finally deteriorates again for the higher layers in the hierarchy. The optimum reconstruction level thus is not obtained at the initial layer (in spite of the fact that this layer by itself provides the maximum amount of information, as seen in Tab. 5.1) but at an intermediate level (layers *pool2* and *pool3*), where already a certain information loss has

taken place. Nevertheless, the features of intermediate complexity seem to be better suited for a statistical pooling.[5]

This observation may be due to two counteracting factors. On the one hand, the spatial resolution in the layers of the network is decreasing with hierarchical level (Tab. 5.1), on the other hand, the number of different features and thereby the size of the statistical representation is increasing. Furthermore, the spatial pooling also causes destructive effects, especially to the spatial information of the features in the image, in addition to its desirable property. However, the increased differentiation capabilities provided by more features in increasing hierarchical levels, may provide a trade-off between these destructive effects and the increased differentiation capabilities [5]. The detailed trade-off properties still have to be researched, however [5]. For example, it has been observed that with a different architecture, which uses large convolution kernels − especially in the first layers − the optimum gets shifted towards the early layers of the network. The true optimum for a statistical representation across *all* possible architectures thus remains to be determined [5].

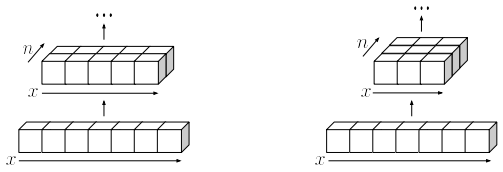


Figure 5.4: Dimensionality reduction in different hierarchical architectures. Spatial resolution is decreased and feature number (and thereby the size of the statistical representation) is increased in both variants, but the trade-off is different. [5]

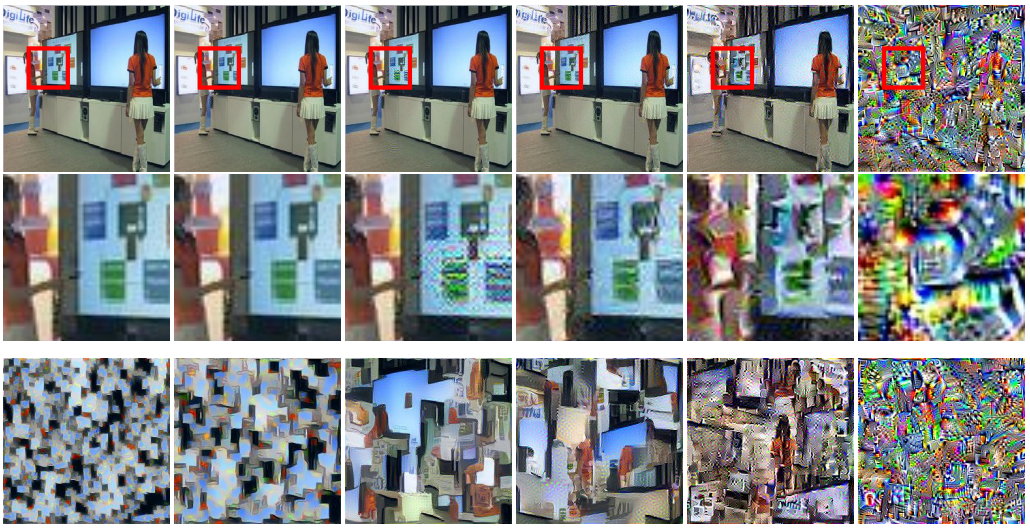


Figure 5.5: Reconstructions from the "raw" feature representation (*upper row*, zoomed in *middle row*) and from the statistical representation (*lower row*). Hierarchical level increases from left to right. Reconstructions are obtained by using only one single layer from the VGG-19 net. [5]

5.6 Peripheral Image Synthesis

This section makes use of the knowledge gained from previous sections in order to implement a model of visual crowding, which is neurobiologically plausible and uses the optimum layer for its statistical representation for the used convolutional network.

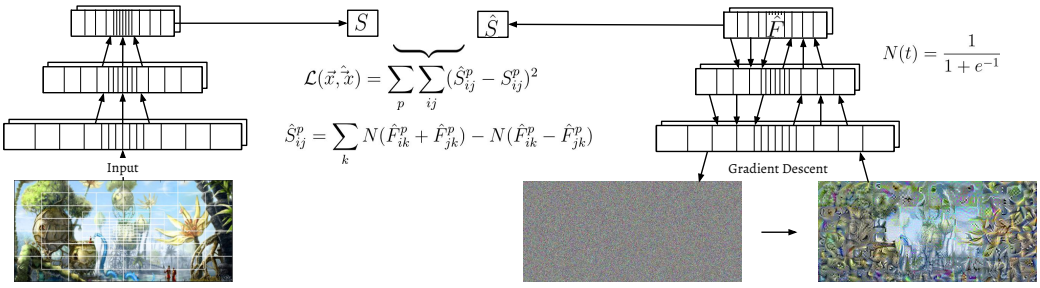


Figure 5.6: Model architecture and reconstruction algorithm. [4]

Our model is illustrated in Fig. 5.6. It differs from the existing crowding model [1, 9] in two aspects: In the use of a deep network architecture and in the provision of neurobiologically plausible AND-like operations as basic non-linearities.

We first apply a spatial tiling operation using the TTM shown in Fig. 5.2 to an image x in which the tiles get larger with increasing eccentricity, roughly similar to the assumed layout of the spatial overlapping pooling regions in human vision, cf. [1, 9]. To aid visual inspection, no smoothing is applied to the tiles, such that borders remain visible. We then compute the activations in layer *pool3* of each vectorized pooling region $\vec{p}$ with a normalized version [11] of the VGG-19 net [36] with average pooling. We use the layer *pool3* due to the results obtained in section "Optimum Layer". These feature maps are then stored in a matrix $F^p \in \mathbb{R}^{N \times M}$, where F^p_{jk} is the activation of the filter j at k in pooling region p, N the number of filter kernels and M the size of each vectorized feature map. In order to obtain a texture-like representation within each pooling region, the spatial information needs to be discarded. Since the AND-like operation in Eq. 5.2 can be viewed as a generalized multiplication [40, 4], we can define the following statistical representation of a generated pooling region $\hat{\vec{p}}$ as

$$\hat{S}^p_{ij} = \sum_k N(\hat{F}^p_{ik} + \hat{F}^p_{jk}) - N(\hat{F}^p_{ik} - \hat{F}^p_{jk}) \tag{5.8}$$

where N is a non-linear transducer function. In our case the sigmoid function.

The loss-function that can be used for the optimization strategy is then defined by

$$\mathcal{L}(\vec{x}, \hat{\vec{x}}) = \sum_p \sum_{ij} (\hat{S}^p_{ij} - S^p_{ij})^2 \tag{5.9}$$

Deriving this equation with respect to the activations $\hat{F}^p$ yields

$$\frac{\partial \mathcal{L}_{ij}^p}{\partial \hat{F}^p} = 2 \sum_k [(\hat{S}^p - S^p)_{ik} \cdot (N'(\hat{F}_{ij}^p + \hat{F}_{kj}^p) - N'(\hat{F}_{ij}^p - \hat{F}_{kj}^p)) + (\hat{S}^p - S^p)_{ki} \cdot (N'(\hat{F}_{kj}^p + \hat{F}_{ij}^p) - N'(\hat{F}_{kj}^p - \hat{F}_{ij}^p))] \tag{5.10}$$

for each pooling region. The loss and gradient for the whole feature space of the image is then summed over the losses and gradients of the single pooling regions.

The gradient $\frac{\partial \mathcal{L}}{\partial \hat{\vec{x}}}$ with respect to the image $\hat{\vec{x}}$ can then be computed via the standard back propagation of the neural network [22], which can then be used as an input for a numerical optimization strategy. Here the the L-BFGS-B solver [42] was used again.

In order to compare the *Neurobiological Model* with the *Mathematical* one, we then computed the mathematical statistics in Eq. 5.3 within each of the pooling regions, as well, and compared both models. The derivative and loss were built accordingly for the Mathematical Model.

5.6.1 Results

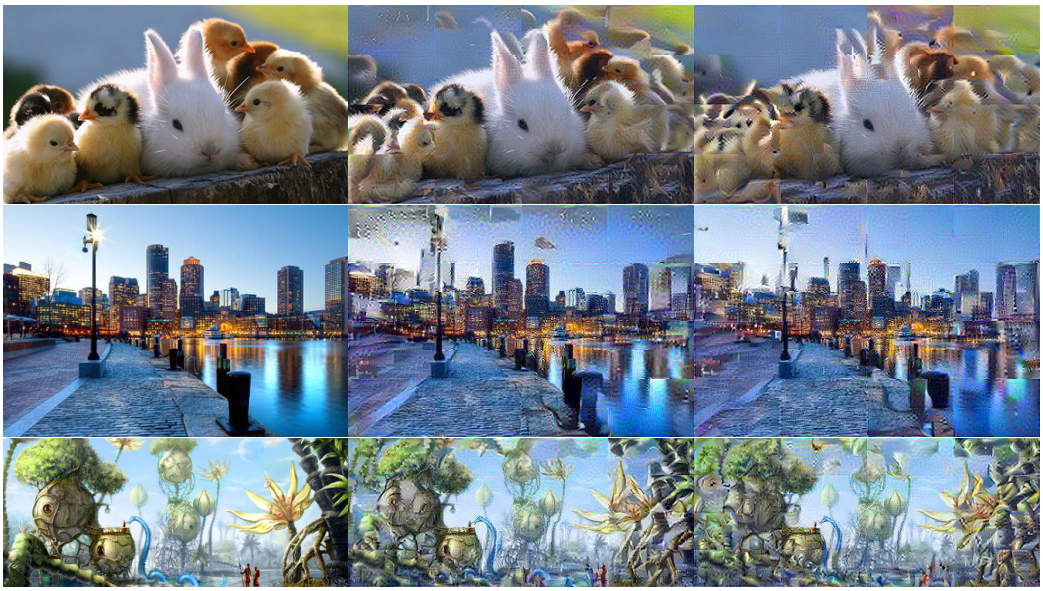


Figure 5.7: Reconstruction results for the *Mathematical* and the *Neurobiological Model*. From left to right: original image, mathematical correlation (Gramian matrix), neurobiological AND-like interactions. Overall reconstruction quality is similar for both models. Detailed visual comparisons can be made by zooming into the visible individual tiles [4]. Image name from top to bottom: *Animals*, *Boston*, *Fantasy*

Model comparison

We compare the *Mathematical* and the *Neurobiological Model* with respect to different criteria.

First, we consider an informal measure by looking at the reconstructed images as such. The *Neurobiological Model* yields similar reconstruction results to the *Mathematical Model*. In some aspects they even appear superior to those obtained with the mathematical correlation statistics (Fig. 5.7).

Additionally, we computed the simple root-mean-square distance from the reconstructed images to the original image (Fig. 5.7, top) and the classifi-

cation performance that can be obtained with the respective reconstruction images using two different classifiers (Fig. 5.7) for a formal measure.

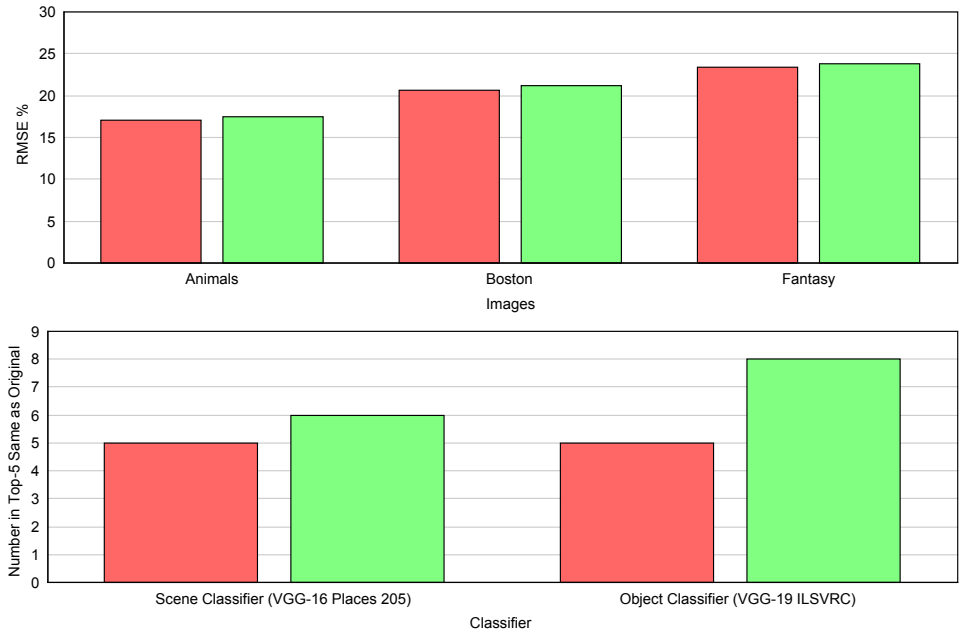


Figure 5.8: Comparisons between mathematical (red) and neurobiological model (green). Upper figure: Similarity of reconstruction results as measured in terms of root-mean-square distance. Lower figure: Similarity of the mathematical and the neurobiological representation as measured in terms of classification performance. Left: place recognition. Right: object recognition. Classification performance is measured in how many of the top-5 classification results were the same as the top-5 results of the original 3 images.

Fig. 5.8 (top) shows that the root-mean-square distance between the reconstructed images and the original images is approximately of the same order for the mathematical models.

Classification performance is measured as the number of top-5 results that were the same in the classification results obtained with the original image and with the reconstructed images. Detailed information about this can be seen in tables in Fig. 5.9 and Fig. 5.10. The images had been classified using the VGG16 net [36] trained on Places205 [41] for scene recognition and

the VGG19 net [36] trained on object recognition using the data from ImageNet2012. Looking at Fig. 5.8 (bottom) also yields that the classification results for both models are similar. Interestingly, the *Neurobiological Model* performs better on the classification performance even though its distance to the original image was always slightly more.
One must consider though, that here just three images were tested due to limited computing resources. So the results may vary a lot from these when using more images to test from but we are optimistic that the results will not change drastically.

In conclusion, our investigations indicate that the information content being represented in the *Neurobiological Model* is comparable or even superior to that of the *Mathematical Model*. This becomes evident in the reconstructed "Mongrel" images, in the distance measures, and in the classification performance. These results may thus be considered as one further step towards a fully plausible model of the neural information processing being performed in the visual periphery. [4]

	Original	Gram	Sigmoid
Animals	Wild Field (23.03%) Aquarium (12.12%) Nursery (7.73%) Wheat Field (6.91%) Pasture (6.10%)	Aquarium (41.92%) Fountain (6.26%) Underwater Coral Reef (4.75%) Outdoor Swimming Pool (4.45%) Amusement Park (4.30%)	Aquarium (62.44%) Underwater Coral Reef (13.64%) Indoor Museum (2.66%) Shower (1.92%) Outdoor Swimming Pool (1.81%)
Boston	Skyscraper (31.24%) Dock (15.47%) Harbor (13.29%) Boardwalk (10.96%) River (10.64%)	Skyscraper (40.23%) Harbor (18.44%) Dock (10.31%) Boardwalk (6.13%) Tower (3.84%)	Skyscraper (28.48%) Harbor (28.02%) Plaza (7.82%) Dock (7.12%) Office Building (3.41%)
Fantasy	Fountain (25.50%) Botanical Garden (13.26%) Swamp (8.77%) Pond (6.37%) Marsh (4.97%)	Corn Field (33.91%) Wheat Field (13.90%) Orchard (7.43%) Indoor Museum (6.31%) Aquarium (6.11%)	Corn Field (51.27%) Aquarium (12.98%) Underwater Coral Reef (6.80%) Swamp (6.69%) Marsh (3.96%)
Compared to Original	-	Animals: 1/5 Boston: 4/5 Fantasy: 0/5 Final: 5/15	Animals: 1/5 Boston: 3/5 Fantasy: 2/5 Final: 6/15
Compared to Gram	-	-	Animals: 3/5 Boston: 3/5 Fantasy: 2/5 Final: 8/15

Figure 5.9: Classification Performance of the original images, compared to the reconstructed ones. The results were obtained from the Places205 CNN.

	Original	Gram	Sigmoid
Animals	Hamster (34.36%) Guinea Pig, Cavia Cobaya (22.25%) Marmoset (12.66%) Hare (9.48%) Hen (4.73%)	Goose (15.75%) Hare (14.95%) King penguin, Aptenodytes Patagonica (14.40%) Fox Squirrel, Eastern Fox Squirrel, Sciurus Niger (8.81%)	Fox Squirrel, Eastern Fox Squirrel, Sciurus Niger (50.32%) Hamster (17.37%) Guinea Pig, Cavia Cobaya (4.03%) Weasel (1.55%) Tiger, Panthera Tigris (1.53%)
Boston	Dock, Dockage, Docking Facility (85.51%) Pier (5.81%) Lakeside, Lakeshore (2.75%) Breakwater, Groin, Groyne, Mole, Bulwark, Seawall, Jetty (1.43%) Seashore, Coast, Seacoast, Sea-Coast (1.25%)	Dock, Dockage, Docking Facility (50.92%) Fountain (18.34%) Church, Church Building (3.95%) Palace (3.32%) Castle (3.17%)	Dock, Dockage, Docking Facility (77.34%) Seashore, Coast, Seacoast, Sea-Coast (4.58%) Pier (2.42%) Breakwater, Groin, Groyne, Mole, Bulwark, Seawall, Jetty (1.95%) Fountain (1.88%)
Fantasy	Fountain (51.93%) Shower Curtain (5.09%) Birdhouse (3.55%) Totem Pole (2.35%) Ear, Spike, Capitulum (1.72%)	Shower Curtain (12.86%) Ear, Spike, Capitulum (4.74%) Fountain (3.86%) Corn (2.48%) Coral Reef (2.47%)	Shower Curtain (23.15%) Fountain (8.29%) Bittern (3.12%) Coral Reef (2.61%) Padlock (2.17%)
Compared to Original	-	Animals: 1/5 Boston: 1/5 Fantasy: 3/5 Final: 5/15	Animals: 2/5 Boston: 4/5 Fantasy: 2/5 Final: 8/15
Compared to Gram	-	-	Animals: 1/5 Boston: 2/5 Fantasy: 3/5 Final: 6/15

Figure 5.10: Classification Performance of the original images, compared to the reconstructed ones. The results were obtained from the VGG19 CNN.

Chapter 6

Summary and Outlook

In recent years, biologically inspired texture-models have provided a fruitful analysis tool for studying visual perception [12]. Especially, the texture-model proposed by Portilla and Simoncelli [29] has sparked a great number of studies in neuroscience and psychophysics [1, 33, 9, 10, 27].

We built a parametric model to synthesize "Mongrels" and simulate peripheral vision and visual crowding with such. This model is based on the local summary statistics of the feature maps of a high-performing deep convolutional neural network. These summary statistics were computed using a more neurobiologically inspired approach by using AND-like operations instead of formal multiplication operations. Furthermore, we determined which layer of the CNN would have the best performing statistical representations. Surprisingly to the common assumption that the first layers would provide the best representation of such, we found an optimum in the intermediate layers of the VGG network. More investigation on this still needs to be done, however.

These results could be of interest in areas like image compression or computer vision. Especially, scene recognition is to be hypothesized to happen

mostly in the visual periphery [8]. Further studies needs to be done on such, as well.

Chapter 7

Peer-Reviewed Publications

Neurobiologically realistic model of statistical pooling in peripheral vision

Noshaba Cheema*, Lex Fridman, Ruth Rosenholtz, and Christoph Zetzsche

University of Applied Sciences Bremen, Germany
Cognitive Neuroinformatics, University of Bremen, Germany
CSAIL, Dept. of Brain and Cognitive Sciences, Massachusetts Institute of Technology, USA

Figure 1: *Reconstruction results for the mathematical and the neurobiological model. From left to right: original image, mathematical correlation (Gram matrix), neurobiological AND-like interactions. Overall reconstruction quality is similar for both models. Detailed visual comparisons can be made by zooming into the visible individual tiles. Image: ©Fantasy Landscape by Deevad*

Keywords: peripheral vision, visual crowding, neuro-biologically motivated statistics, neural networks, image compression

Concepts: •**Mathematics of computing** → Probability and statistics;

1 Introduction

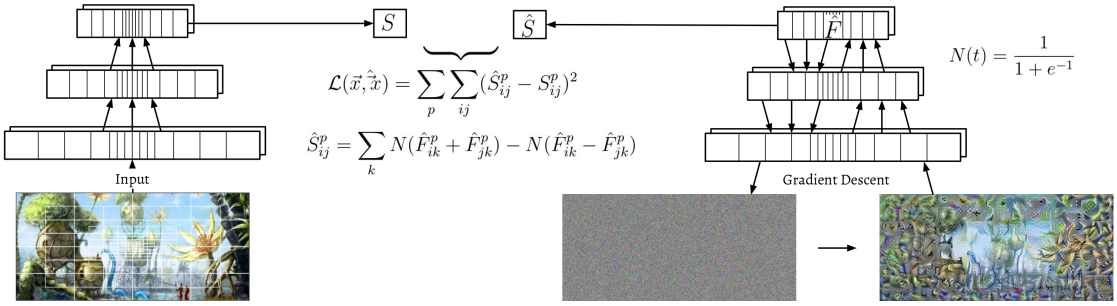
Our senses can process only a limited amount of the incoming sensory information. In the human visual system this becomes apparent in the reduced performance in the peripheral field of view, as compared to the central fovea. Recent research shows that this loss of information cannot solely be attributed to a spatially coarser resolution but is essentially caused by statistical pooling operations which lead to a texture-like representation. This so called "crowding effect" can be seen as a strategy of the visual system to deal with the information bottleneck in sensory processing. As such, it may also be of interest for image compression, feature extraction and computer vision. A recent approach towards modelling this effect makes direct use of formal statistical computations [Balas et al. 2009; Freeman and Simoncelli 2011], and thus is not completely convincing with respect to its neurobiological plausibility. Here we investigate whether a more plausible model can be developed and whether it achieves the desirable properties.

2 Summary statistics model

Our model is illustrated in Fig. 2. It differs from the existing crowding model [Balas et al. 2009; Freeman and Simoncelli 2011] in two aspects, in the use of a deep network architecture and in the provision of neurobiologically plausible AND-like operations as basic nonlinearities. We first apply a spatial tiling operation to an image x in which the tiles get larger with increasing eccentricity, roughly similar to the assumed layout of the spatial overlapping pooling regions in human vision, cf. [Balas et al. 2009; Freeman and Simoncelli 2011]. To aid visual inspection, no smoothing is applied to the activation of the filter j at k in pooling region p, N the number of filter kernels and M the size of each vectorized feature map.

In order to obtain a texture-like representation within each pooling region, the spatial information needs to be discarded. A summary statistic that does this, is given by the correlations $S^p = \sum_k F^p_{ik} F^p_{jk}$ based on the feature map matrix F of a pooling region p. Like the crowding model of [Balas et al. 2009; Freeman and Simoncelli 2011] the present model version makes explicit use of the *multiplication* of two variables. However, researchers have long debated whether such multiplication operations are biologically plausible [Koch 2004]. Furthermore, computation of the formal statistical correlations may not be necessary, as much of the functionality may be preserved if multiplication is replaced by a neurobiologically more plausible operation. We thus replace the multiplications by neurophysiologically plausible AND-like operations [Zetzsche and Barth 1990] . It is well known that biological hardware can easily realize an ON/OFF rectification and nonlinear transducer functions with sigmoid shape. With these ingredients, one can make use of an old Babylonian trick to derive the suitable AND-like computations: $AND(a, b) = N[a + b] - N[a - b]$ where N is a suitable nonlinear transducer function. These AND operations are characterized by the property that they attain their maximum (for the sum $a + b$ constrained) if a and b have the same size. If a or b is decreased, the response is systematically reduced, until it vanishes if either a or b equals zero. If N is a sigmoid nonlinearity the resulting AND will have a threshold-like behavior for small input values and will go into saturation for large input values. For simplicity, the model version which is based on the formal statistical computations is henceforth designated as "mathematical model" (in spite of its other neurobiology-related components) and the model version with the AND operations is designated as "neurobiological model".

The information provided by the different model representations can be visualized and evaluated by reconstructed images, designated as *mongrels* [Balas et al. 2009; Rosenholtz et al. 2012] and as



$$\mathcal{L}(\vec{x},\hat{\vec{x}}) = \sum_p \sum_{ij} (\hat{S}^p_{ij} - S^p_{ij})^2$$

$$\hat{S}^p_{ij} = \sum_k N(\hat{F}^p_{ik} + \hat{F}^p_{jk}) - N(\hat{F}^p_{ik} - \hat{F}^p_{jk})$$

$$N(t) = \frac{1}{1+e^{-1}}$$

Figure 2: *Model architecture and reconstruction algorithm.*

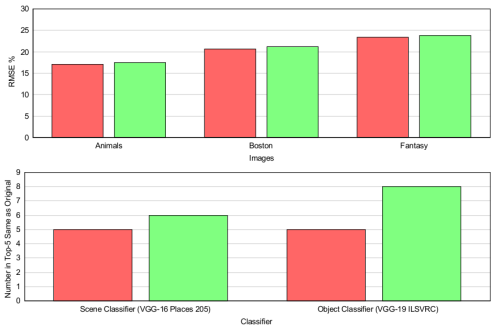


Figure 3: *Comparisons between mathematical (red) and neurobiological model (green). Upper figure: Similarity of reconstruction results as measured in terms of root-mean-square distance. Lower figure: Similarity of the mathematical and the neurobiological representation as measured in terms of classification performance. Left: place recognition. Right: object recognition. Classification performance is measured in how many of the top-5 classification results were the same as the top-5 results of the original 3 images.*

that is used for this optimization strategy is defined by $\mathcal{L}(\vec{x},\hat{\vec{x}}) = \sum_p \sum_{ij} (\hat{S}^p_{ij} - S^p_{ij})^2$.

3 Model Comparison

We compare the mathematical and the neurobiological with respect to different criteria. First, we consider the reconstructed images as such. The neurobiological model yields reconstruction results which are at least as good, and in some aspects even superior to those obtained with the mathematical correlation statistics (Fig. 1). Further measures of the similarity between the reconstructions obtained from the two types of models are the simple root-mean-square deviation from the original image, and the classification performance that can be obtained with the respective reconstruction images. Fig. 3 shows that the root-mean-square distance betwwen the reconstructed images and the original images is approximately of the the same order for the mathematical models and the neurobiological models and that the classification performance is also similar. The images had been classified using the VGG16 net trained on Places205 for scene recognition and the VGG19 net trained on object recognition using the data from ImageNet2012. A more detailed table can be found here.

In conclusion, our investigations indicate that the information content being represented in the neurobiological model is comparable or even superior to that of the the mathematical model. This becomes evident in the reconstructed "Mongrel" images, in the distance measures, and in the classification performance. These results may thus be considered as one further step towards a fully plausible model of the neural information processing being performed in the visual periphery.

References

BALAS, B., NAKANO, L., AND ROSENHOLTZ, R. 2009. A summary-statistic representation in peripheral vision explains visual crowding. *Journal of vision 9*, 12, 13–13.

FREEMAN, J., AND SIMONCELLI, E. P. 2011. Metamers of the ventral stream. *Nature neuroscience 14*, 9, 1195–1201.

GATYS, L., ECKER, A. S., AND BETHGE, M. 2015. Texture synthesis using convolutional neural networks. In *Advances in Neural Information Processing Systems*, 262–270.

KOCH, C. 2004. *Biophysics of computation: information processing in single neurons*. Oxford university press.

PORTILLA, J., AND SIMONCELLI, E. P. 2000. A parametric texture model based on joint statistics of complex wavelet coefficients. *International Journal of Computer Vision 40*, 1, 49–70.

ROSENHOLTZ, R., HUANG, J., RAJ, A., BALAS, B. J., AND ILIE, L. 2012. A summary statistic representation in peripheral vision explains visual search. *Journal of vision 12*, 4, 14–14.

SIMONYAN, K., AND ZISSERMAN, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

ZETZSCHE, C., AND BARTH, E. 1990. Fundamental limits of linear filters in the visual processing of two-dimensional signals. *Vision research 30*, 7, 1111–1117.

Optimum statistical representation obtained from an intermediate feature level of the visual hierarchy.

Noshaba Cheema[1,2], Lex Fridman[3], Ruth Rosenholtz[3], and Christoph Zetzsche[1⋆]

[1] Cognitive Neuroinformatics, University of Bremen, Germany
[2] University of Applied Sciences Bremen, Germany
[3] CSAIL, Dept. of Brain and Cognitive Sciences, MIT, USA

Abstract. Representations obtained from the statistical pooling of features gain increasing popularity. The common assumption is that low-level features are best suited for such a statistical pooling. Here we investigate which level of a visual feature hierarchy can actually produce the optimal statistical representation. We make use of the award-winning VGG 19 deep network which showed human-like performance in recent visual recognition benchmarks. We demonstrate that the optimum statistical representation is not obtained with the early-level features, but with those of intermediate complexity. This could provide a new perspective for models of human vision, and could be of general relevance for statistical pooling approaches in computer vision and image processing.

1 Introduction

Representations that are based on a statistical pooling of features are of relevance in variety of contexts, ranging from bag-of-words models in natural language processing to texton approaches in computer vision [2, 5]. In research on human vision, such a statistical pooling has recently been recognized as the crucial factor for the representation of information in the periphery of the field of view, i.e. for those 98% of total area being not represented by the high-performance central fovea [1, 3]. A common assumption about statistical pooling is that the features to be pooled should be relatively simple and low-level. In vision and image processing, for example, local wavelet-like features with different orientations and sizes are commonly utilised. However, convincing as this assumption may appear on a first look, it is actually far from clear why low-level features should be best suited for a statistical pooling. Multi-level feature hierarchies have proven useful in a variety of contexts. The visual system, for example, consists of a hierarchy of multiple subsequent processing stages in which the nature of the visual features becomes systematically more abstract, invariant, and general. The impressive success of recent state-of-the-art *deep networks*, which for the first

⋆ corresponding author: zetzsche@informatik.uni-bremen.de

2

time challenged human performance in visual recognition benchmarks [6], seems also to indicate that such hierarchical architectures are crucial for obtaining efficient representations. However, which level in such hierarchy is best suited for a statistical pooling is in our view far from obvious. Here we investigate this question by making use of an award-winning deep architecture, the 19 layer *VGG 19* net [6].

2 Methods

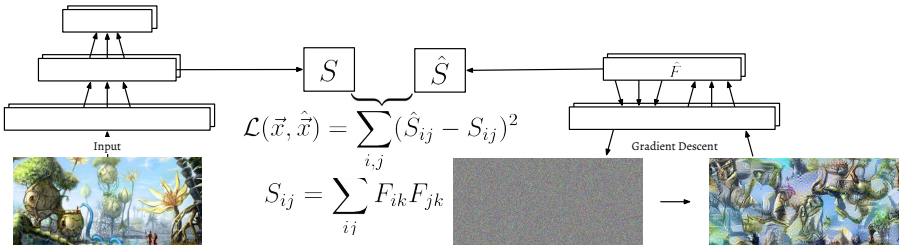


Fig. 1: Schematic architecture of the deep network and reconstruction algorithm. (F_{ik} : feature i at position k; S_{ij} : covariance of features F_i, F_j)

The architecture of the model is shown in Fig. 1. The feature hierarchy is provided by the 19 layer VGG-19 network, as schematically illustrated on the left side. For the statistical pooling part we use a recent extension of this architecture intended for texture synthesis [4]. This texture synthesis procedure has the goal to generate a texture which has the same statistical properties as a given example texture. Here we make use of this process with a somewhat different conceptual attitude. As suggested by Rosenholtz [1], we use this ability to *reconstruct* a general image from its statistical representation [1, 3]. In the original texture synthesis model, the statistical information is accumulated across layers, starting from the first layer and comprising all subsequent layers up to some specified maximum height in the hierarchy. We took a different approach here, and selected out only one single layer of the VGG-19 net in each test, since we wanted to know which layer of the hierarchy can provide the most valuable information for a statistical representation. This is schematically illustrated for the second layer in Fig. 1. We reconstruct the image from solely the statistical information S, which is provided by the covariance/spatial pooling of all respective features F_{ik}, F_{jk} of this layer (i.e, we reconstruct from the *statistical representation* provided by the box S in Fig. 1). For comparison, we also applied the reconstruction procedure to the single "raw" features F_{ik} of this layer (*feature representation*, leftmost side of Fig. 1). In other words, in the case of the "raw" *feature representation* we reconstruct directly from the coefficients in

a single layer of the hierarchy, whereas in case of the *statistical representation* we reconstruct from the covariance/spatial pooling of those coefficients.

3 Results

A prototypical result is shown in Fig. 2. As expected, the best reconstruction from the "raw" feature representation is obtained from the initial, lowest level of the network. The reconstructions then become gradually more and more distorted if we proceed towards higher layers (from left to right in Fig. 2, upper two rows). This is not surprising, since the amount of information is systematically decreased by the increasing abstraction towards higher stages of the hierarchy. The quality of reconstructions from the statistical representation is generally lower, due to the statistical pooling. However, a monotonic decrease of reconstruction quality is not observed here. Rather, the reconstruction quality is low at the initial stage, then gradually increases up to the intermediate layers, and finally deteriorates again for the higher layers in the hierarchy. The optimum reconstruction thus is not obtained at the initial layer (in spite of the fact that this layer by itself provides the maximum amount of information) but at an intermediate level (layers *pool2* and *pool3*) where already a certain information loss has taken place. Nevertheless, the features of intermediate complexity seem to be better suited for a statistical pooling.

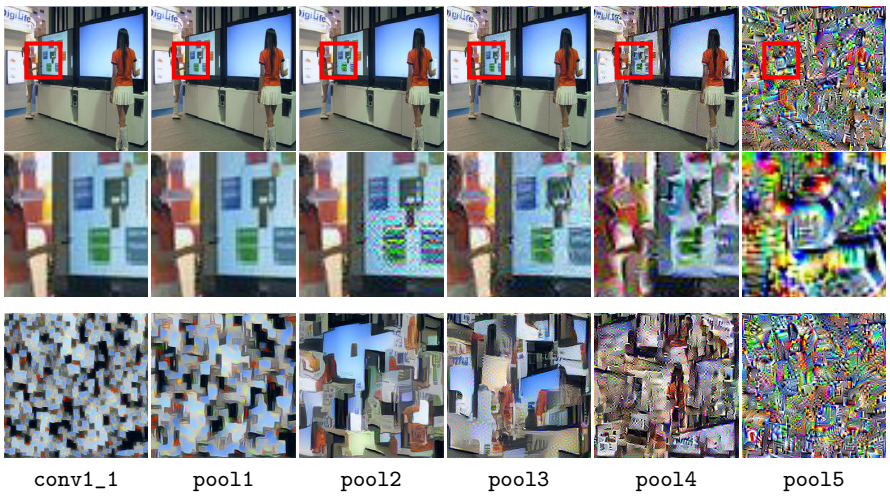


Fig. 2: Reconstructions from the "raw" feature representation (*upper row*, zoomed in *middle row*) and from the statistical representation (*lower row*). Hierarchical level increases from left to right. Reconstructions are obtained by using only one single layer from the VGG-19 net.

4

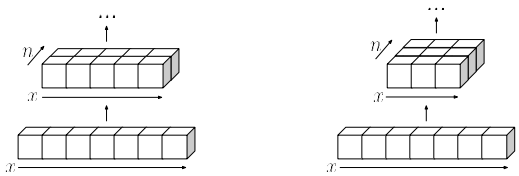


Fig. 3: Dimensionality reduction in different hierarchical architectures. Spatial resolution is decreased and feature number (and thereby the size of the statistical representation) is increased in both variants, but the tradeoff is different.

4 Discussion

Why are the optimum features for a statistical representation those from an intermediate level, and not the features of the initial processing stage which provides the highest information content? This may be due to two counteracting factors (Fig. 3, left). On the one hand, the total number of coefficients in the layers of the network is decreasing with increasing hierarchical level. On the other hand, the number of different features, and thereby the size of the statistical representation is increasing. Since the statistical spatial pooling causes specific destructive effects, in particular with respect to the exact position of features in the image, the trade-off between these effects and the increased differentiation capabilities provided by more different features may lead to an optimum at an intermediate stage. The detailed trade-off, however, depends on the specific architecture (cf. Fig. 3). For example, in preliminary tests with a network with large convolution kernels and a higher number of different features in the early layers, we observed a shift of the optimum towards the early stages. The true optimum for a statistical representation across *all* possible architectures thus remains to be determined. In any case our results suggest that the exploitation of the mutual dependence between the abstraction effect in feature hierarchies on the one hand, and the peculiarities of statistical measures and pooling effects on the other hand, represents a promising topic for future research.

References

1. Balas, B., Nakano, L., Rosenholtz, R.: A summary-statistic representation in peripheral vision explains visual crowding. Journal of vision 9(12), 13–13 (2009)
2. Cambria, E., White, B.: Jumping nlp curves: a review of natural language processing research. Computational Intelligence Magazine, IEEE 9(2), 48–57 (2014)
3. Freeman, J., Simoncelli, E.P.: Metamers of the ventral stream. Nature neuroscience 14(9), 1195–1201 (2011)
4. Gatys, L., Ecker, A.S., Bethge, M.: Texture synthesis using convolutional neural networks. In: Advances in Neural Information Processing Systems. pp. 262–270 (2015)
5. Renninger, L.W., Malik, J.: When is scene identification just texture recognition? Vision research 44(19), 2301–2311 (2004)
6. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)

Bibliography

[1] Balas, B., Nakano, L., and Rosenholtz, R. A summary-statistic representation in peripheral vision explains visual crowding. *Journal of vision 9*, 12 (2009), 13.1–18.

[2] Blasdel, G. G., and Fitzpatrick, D. Physiological organization of layer 4 in macaque striate cortex., 1984.

[3] Carandini, M., Demb, J. B., Mante, V., Tolhurst, D. J., Dan, Y., Olshausen, B. A., Gallant, J. L., and Rust, N. C. Do we know what the early visual system does? *The Journal of neuroscience : the official journal of the Society for Neuroscience 25*, 46 (2005), 10577–97.

[4] Cheema, N., Fridman, L., Rosenholtz, R., and Zetzsche, C. Neurobiologically realistic model of statistical pooling in peripheral vision. *Submitted for publication*.

[5] Cheema, N., Fridman, L., Rosenholtz, R., and Zetzsche, C. Optimum statistical representation obtained from an intermediate feature level of the visual hierarchy. *Submitted for publication*.

[6] Ciresan, D., Meier, U., Masci, J., Gambardella, L. M., and Schmidhuber, J. Flexible, High Performance Convolutional Neural Networks for Image Classification. *International Joint Conference on Artificial Intelligence (IJCAI) 2011* (2011), 1237–1242.

[7] Dalal, N., and Triggs, B. Histograms of Oriented Gradients for Human Detection. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) 1* (2005), 886–893.

[8] Eberhardt, S. *Analysis and Modeling of Visual Invariance for Object Recognition and Spatial Cognition*. PhD thesis, University of Bremen, 2015.

[9] Freeman, J., and Simoncelli, E. P. Metamers of the ventral stream. *Nature neuroscience 14*, 9 (2011), 1195–1201.

[10] Freeman, J., Ziemba, C. M., Heeger, D. J., Simoncelli, E. P., and Movshon, J. A. A functional and perceptual signature of the second visual area in primates. *Nature neuroscience 16*, 7 (2013), 974–81.

[11] Gatys, L. A., Ecker, A. S., and Bethge, M. A Neural Algorithm of Artistic Style. *arXiv preprint* (2015), 3–7.

[12] Gatys, L. A., Ecker, A. S., and Bethge, M. Texture Synthesis Using Convolutional Neural Networks. *NIPS* (2015), 1–10.

[13] Goodale, M., and Milner, A. Separate visual pathways for perception and action. *Trends in Neurosciences 15*, 1 (1992), 20–25.

[14] Hannun, A., Case, C., Casper, J., Catanzaro, B., Diamos, G., Elsen, E., Prenger, R., Satheesh, S., Sengupta, S., Coates, A., and Ng, A. Y. Deep Speech: Scaling up end-to-end speech recognition. *Arxiv* (2014), 1–12.

[15] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. Caffe: Convolutional Architecture for Fast Feature Embedding. In *Proceedings of the ACM International Conference on Multimedia* (2014), pp. 675–678.

[16] Johnson, D. Composing Music With Recurrent Neural Networks, 2015.

[17] Julesz, B. Textons, the elements of texture perception, and their interactions. *Nature 290*, 5802 (1981), 91–97.

[18] Karpathy, A., and Fei-Fei, L. Deep visual-semantic alignments for generating image descriptions. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on* (2015), pp. 3128–3137.

[19] Karpathy, A., Johnson, J., and Fei-Fei, L. Visualizing and Understanding Recurrent Networks. *arXiv* (jun 2015), 1–13.

[20] Koch, C. *Biophysics of Computation: Information Processing in Single Neurons*, vol. 11. 2004.

[21] Krizhevsky, A., Sutskever, I., and Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks. *Advances In Neural Information Processing Systems* (2012), 1–9.

[22] LeCun, Y. A., Bottou, L., Orr, G. B., and M??ller, K. R. Efficient backprop. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 7700 LECTU* (2012), 9–48.

[23] Lowe, D. G. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision* (1999), vol. 2, IEEE, pp. 1150–1157.

[24] Mitchell, T. M. *Machine Learning*. No. 1. 1997.

[25] Mordvintsev, A., Olah, C., and Tyka, M. Inceptionism: Going Deeper into Neural Networks, 2015.

[26] Movshon, J. a., Adelson, E. H., Gizzi, M. S., and Newsome, W. T. The analysis of moving visual patterns, 1985.

[27] Okazawa, G., Tajima, S., and Komatsu, H. Image statistics underlying natural texture selectivity of neurons in macaque V4. *Proceedings of the National Academy of Sciences of the United States of America 112*, 4 (2015), E351–60.

[28] Pelli, D. G., and Tillman, K. a. The uncrowded window of object recognition. *Nature neuroscience 11*, 10 (2008), 1129–1135.

[29] Portilla, J., and Simoncelli, E. P. Parametric texture model based on joint statistics of complex wavelet coefficients. *International Journal of Computer Vision 40*, 1 (2000), 49–71.

[30] Roe, A. W., Chelazzi, L., Connor, C. E., Conway, B. R., Fujita, I., Gallant, J. L., Lu, H., and Vanduffel, W. Toward a Unified Theory of Visual Area V4, 2012.

[31] Rojas, R. Neural networks: a systematic introduction. *Neural Networks* (1996), 502.

[32] Rosenholtz, R. What your visual system sees where you are not looking. *Proc SPIE Human Vision and Electronic Imaging 7*, 1 (2011), 786510–786510–14.

[33] Rosenholtz, R., Huang, J., Raj, a., Balas, B. J., and Ilie, L. A summary statistic representation in peripheral vision explains visual search. *Journal of Vision 12*, 4 (2012), 14–14.

[34] Schmidhuber, J. Deep Learning in Neural Networks: An Overview. *arXiv preprint arXiv:1404.7828* (2014), 1–66.

[35] Simoncelli, E. P., and Freeman, W. T. The Steerable Pyramid: A Flexible Multi-Scale Derivative Computation. *Conference, Ieee International Processing, Image (Rochester, N.Y.) III* (1995), 444–447.

[36] Simonyan, K., and Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *ArXiv* (2014), 1–14.

[37] Strasburger, H., Rentschler, I., and Jüttner, M. Peripheral vision and pattern recognition: a review. *Journal of vision 11*, 5 (2011), 13.

[38] Sutton, R. S., and Barto, A. G. Introduction to Reinforcement Learning. *Learning 4*, 1996 (1998), 1–5.

[39] Ungerleider, L. G., and Mishkin, M. Two cortical visual systems. In *Analysis of Visual Behavior*. 1982, pp. 549–586.

[40] Zetzsche, C., and Barth, E. Fundamental limits of linear filters in the visual processing of two-dimensional signals, 1990.

[41] Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., and Oliva, A. Learning Deep Features for Scene Recognition using Places Database. *Advances in Neural Information Processing Systems 27* (2014), 487–495.

[42] Zhu, C., Byrd, R. H., Lu, P., and Nocedal, J. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software 23*, 4 (1997), 550–560.